LaTeX 下的页面布局*

Piet van Oostrum†
Dept. of Computer Science
Utrecht University

2001 年 8 月 12 日

摘 要

该篇文章主要描述了如何在你的 LaTeX 文档中调整页面布局，也就是如何改变页边距和页面大小、页眉页脚，以及图片或表格（统称浮动对象）的适当位置。

本文原来为 fancyheadings 宏包的文档。当然也包括其他信息，比如标记（marks）的高级使用方法，以及处理浮动对象的方法。附带在 fancyheadings 宏包中的文档已经升级为版本2。考虑到各个操作系统间的兼容性，宏包的名字已经改成了 fancyhdr。

目 录

*译者：ifuleyou，bbs.ctex.org。希望大家不吝指正，排版翻译上的错误我都要！

†这份文档中的相当一部分是由 George Grätzer (University of Manitoba) 在*Notices Amer. Math. Soc.*撰写。谢谢你，George！

1 简介

在 LaTeX 文档中，页面是由各种不同的元素组合而成，如插图1 所示。 Body 中包含了文档的正文以及浮动对象（表格和图片）。

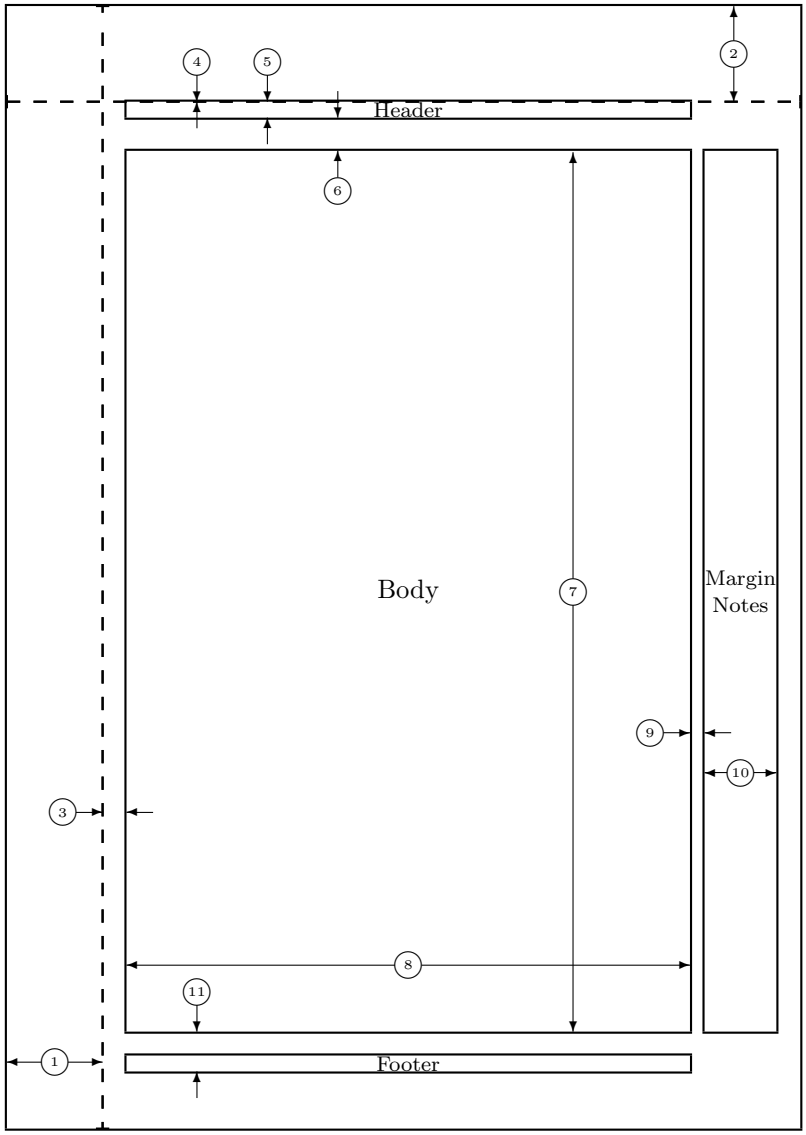
整个页面是由 LaTeX 的输出过程 (Output Routine) 来构造的，该输出过程很复杂，因此不宜对其进行修改。本文中所描述的一些宏包对输出过程进行了少许的修改以便达到一些其他方法所无能为力的效果。你最好使用这些宏包而尽量避免自己去盲目修改输出过程。

有些东西必须在这里向你说明一下：

1. 左边距不叫 `\leftmargin`，而是 `\evensidemargin`（偶数页面中）和 `\oddsidemargin`（奇数页面中）。在单面的文档中，`\oddsidemargin`就代表了两种边距。`\leftmargin` 也是个 LaTeX 命令，但它有不同的含义（就是列表的缩进距离）。
2. 大部分参数不适合在文档中间进行修改，而另一些修改可能在分页的时候有效果。如果你只想改变该页的高度，你可以使用 `\enlargethispage`命令。

边注区（margin notes area）包含了一小段由命令 `\marginpar` 所生成的文本。在双开面的文档中，边注区交替出现在左右两边。边注区的位置并不根据页面大小而固定不变，而是和边注所在的段落具有差不多的高度。由于计算边注区位置的算法的关系，在双开面的文档中，如果边注比较靠近分页的地方，那么它很可能会出现在错误的那一边。如果你想在特定的地方放置你的边注，你可以参考 19 节或是 20 节中的内容。

本文的第一部分将描述如何改变页眉以和页脚区，而最后一部分将描述如何将你的浮动对象放置在想要的地方。



1	one inch + \hoffset	2	one inch + \voffset
3	\oddsidemargin = 18pt	4	\topmargin = 0pt
5	\headheight = 12pt	6	\headsep = 25pt
7	\textheight = 663pt	8	\textwidth = 424pt
9	\marginparsep = 11pt	10	\marginparwidth = 54pt
11	\footskip = 30pt		\marginparpush = 5pt (not shown)
	\hoffset = 0pt		\voffset = 0pt
	\paperwidth = 597pt		\paperheight = 845pt

图 1: 页面元素。这些数值仅仅反映本文档中的元素，并不代表默认的值。

2 页眉和页脚

在 LaTeX 中，页眉和页脚的样式是由命令 \pagestyle 和 \pagenumbering 来定义的。\pagestyle命令定义了页眉和页脚的基本内容（如页码出现在哪里），而 \pagenumbering 则定义了页码的显示方式。LaTeX 本身包含四种标准的页面样式。

empty	没有页眉也没有页脚
plain	没有页眉，页脚包含一个居中的页码
headings	没有页脚，页眉包含章/节或者子节的名字和页码
myheadings	没有页脚，页眉包含有页码和用户提供的其他信息

尽管这些样式相当有用，但功能比较有限。其他页面样式可以通过 \ps@xxx这些命令来定义。当文档中遇到命令 \pagestyle{xxx}时，这些定义被执行。命令 \ps@xxx 应该为页眉和页脚定义如下命令：

\@oddhead	双开面文档中，奇数页面的页眉（单开面中为所有页面的页眉）
\@evenhead	双开面文档中，偶数页面的页眉
\@oddfoot	双开面文档中，奇数页面的页脚（单开面中为所有页面的页脚）
\@evenfoot	双开面文档中，偶数页面的页脚

这些并不是命令，而是 LaTeX 的输出过程所使用的“变量”。这些含有字符 ‘@’ 的命令或变量最好只出现在宏包中，或者位于命令 \makeatletter 和 \makeatother 之间。

命令 \pagenumbering 定义了页码的形式。它带有一个参数，为以下几种之一：

arabic	阿拉伯数字
roman	小写的罗马数字
Roman	大写的罗马数字
alph	小写字母
Alph	大写字母

命令 \pagenumbering{xxx} 定义了另外一个命令 \thepage ，使得该命令扩展之后成为 xxx 的形式。然后 \pagestyle 命令便将 \thepage 放在适当的位置。另外，\pagenumbering 命令同时也将把页码重置为1。\pagestyle 和 \pagenumbering 将影响到当前正在构建的页面，因此最好将它们放在想要它们起作用的页面的位置上（参考 17 节）。

3 fancyhdr 是什么？

宏包 fancyhdr 可以让你方便地调整 LaTeX 文档中的页眉和页脚。你可以定义：

- 三部分组成的页眉和页脚
- 页眉和页脚中的修饰线
- 可以比正文更宽的页眉和页脚
- 多行的页眉和页脚
- 对应于奇数和偶数页面不同的页眉和页脚
- 章节起始页的页眉和页脚可以和其他页面不同
- 包含浮动对象的页面其页眉和页脚可以和其他页面不同

当然，你完全可以对字体、大小写等进行控制。

4 fancyhdr 的简单应用

在 LaTeX 2ε 文档中使用该宏包，你必须将文件fancyhdr.sty 放在 TeX 可以找到的目录或文件夹内（通常在输入目录），然后在你的导言区

```
\documentclass{...}
```

后加入命令[1]：

```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

我们看到通过 fancyhdr 构造的页面如下：

LeftHeader	CenteredHeader	RightHeader
	page body	
LeftFooter	CenteredFooter	RightFooter

LeftHeader 和 LeftFooter 为居左对齐；CenteredHeader 和 CenteredFooter 居中；而 RightHeader 和 RightFooter 居右对齐。

六个部分和两条修饰线可以各自单独定义。

5 举个简单的例子

K. Grant 正在给校长 A. Smith 写一份报告，是关于“新毕业学生的表现”，他采用了如下的页面布局：

		新毕业学生的表现
	page body	
From: K. Grant	To: Dean A. Smith	3

“3”是页码，标题“新毕业学生的表现”为黑体。

这些效果是由 \pagestyle{fancy}[2] 之后的这些命令所获得：

```
\lhead{}
\chead{}
\rhead{\bfseries 新毕业生的表现}
```

[1]对于LaTeX2.09版本，你必须在命令 \documentstyle 中指定使用 [fancyhdr] 而不是用 \usepackage 命令。
[2]注意：在 fancyheadings 的版本1中，是通过 \setlength 来改变 \...rulewidth 参数的。

```
\lfoot{From: K. Grant}
\cfoot{To: Dean A. Smith}
\rfoot{\thepage}
\renewcommand{\headrulewidth}{0.4pt}
\renewcommand{\footrulewidth}{0.4pt}
```

（命令 `\thepage` 用于显示当前页码，而 `\bfseries` 是 LaTeX 2$_\varepsilon$ 用于选择黑体的命令[3]）

看上去挺不错，但第一页显然不需要这些页眉和页脚。在 `\begin{document}` 之后而在 `\maketitle` 命令之前敲入下面的命令来去掉页眉和页脚中其他内容而只保留页码：

```
\thispagestyle{plain}
```

或者

```
\thispagestyle{empty}
```

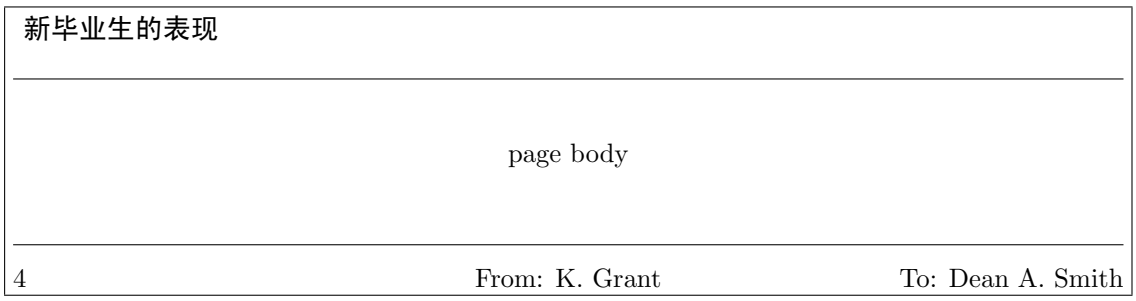
如果你什么也不想要的话。

实际上，LaTeX 命令 `\maketitle` 中已经定义了命令 `\thispagestyle{plain}`。因此，如果你坚持要在 `\maketitle` 生成的页面上使用 fancy 的布局，你必须在 `\maketitle` 之后马上发出`\thispagestyle{fancy}` 的指示。

6 双面打印的例子

一些文档类如 `book.cls`，默认为双面排版：单双数页面有不同的布局。另外一些文档类通过选项 `twosides` 来取得双面效果。

现在我们用双面重新排版报告，上面一节中的布局留给奇数页面，而对于偶数页面（左页面）采用如下布局：



“4”为页码。

下面是相应的命令：

```
\fancyhead{} % clear all fields
\fancyhead[RO,LE]{\bfseries 新毕业生的表现}
\fancyfoot[LE,RO]{\thepage}
\fancyfoot[LO,CE]{From: K. Grant}
\fancyfoot[CO,RE]{To: Dean A. Smith}
\renewcommand{\headrulewidth}{0.4pt}
```

[3]译者：在 CJK 中可以直接用 `\bfseries` 来改变中文字体，而在 CCT 中则需要 `\ziti` 来完成。

E	偶数页
O	奇数页
L	居左内容
C	居中内容
R	居右内容
H	页眉
F	页脚

图 2: Selectors

```
\renewcommand{\footrulewidth}{0.4pt}
```

我们使用了更通用的命令 \fancyhead 和 \fancyfoot。这两个命令可以通过参数指定奇/偶数页的页眉/页脚的哪个部分采用何种格式。其中，第一个命令忽略了这些参数，从而指定为针对任何页眉。通常，这在覆盖默认或者之前定义的时候比较有用。方括号中的选择参数由图 2 列出。选择参数可以组合，比如 \fancyhead[LE,RO]{text} 就定义了偶数页的左页眉和奇数页的右页眉。如果你没有给出 E 或 O 则将针对所有页面。对于 LRC 也是同样道理。因此前面的 \lhead 只是 \fancyheadd[L] 命令的缩写罢了。选择参数并不区分大小写。

更通用一点，通过选择参数H (header) 和 F (footer) 命令 \fancyhf 可以让你同时指定页眉和页脚。实际上，\fancyhead 和 \fancyfoot 也不过是指定了 H 和 F 的 \fancyhf 命令而已。

再说一句，你可以通过 \thispagesytle{plain} 来定义第一页的简单布局。

7 重新定义 plain 样式

一些 LaTeX 命令，像 \chapter，使用命令 \thispagestyle 来切换到 plain 样式，因此你自己定义的样式不起作用。要调整这种页面的布局，你必须重新定义 plain 样式。正如开头说的那样，你可以重新定义\ps@plain 命令，但 fancyhdr 提供了一种更简单的方法。命令 \fancypagestyle 可以重新定义现有的页面样式（如 plain）或者新的页面样式，它有两个参数：第一个是定义的页面样式的名字，第二个参数包含了改变页眉页脚的那些命令，如 fancyhead 等。同样，\headrulewidth 和 \footrulewidth 也可以改变。举个例子，我们为 6节中的报告重新定义 plain 样式，使得其页码为黑体：

```
\fancypagestyle{plain}{%
\fancyhf{} % clear all header and footer fields
\fancyfoot[C]{\bfseries \thepage} % except the center
\renewcommand{\headrulewidth}{0pt}
\renewcommand{\footrulewidth}{0pt}}
```

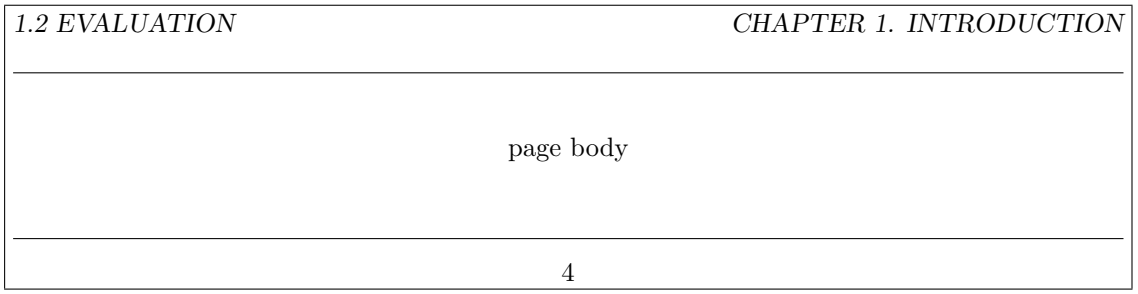
8 默认布局

如果我们只使用 book.cls 文档类以及 fancyhdr 的默认设置，则只需要如下命令：

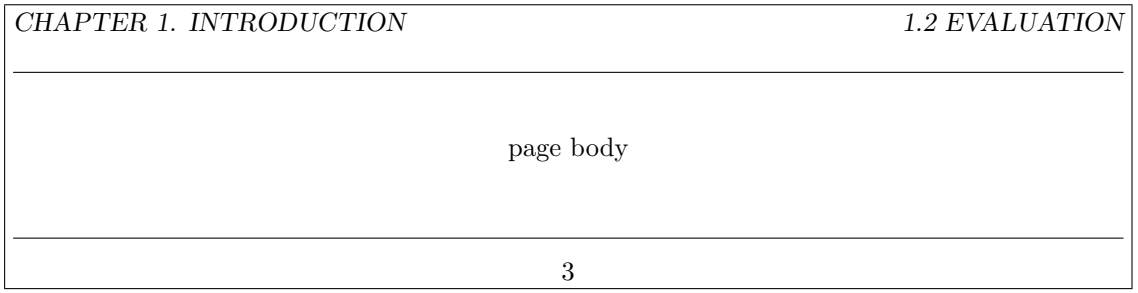
```
\usepackage{fancyhdr}
\pagestyle{fancy}
```

fancyhdr 将会包揽全部工作。在新的一章开始的页面上，页码出现在页脚的中间，没有页眉也没有修饰线。

在偶数页面上，我们有如下布局：



在奇数页面上，我们得到：



页眉的文字大写并倾斜

默认的布局由下面的命令构成：

```
\fancyhead[LE,RO]{\slshape \rightmark}
\fancyhead[LO,RE]{\slshape \leftmark}
\fancyfoot[C]{\thepage}
```

修饰线则为如下的设置：

`\headrulewidth`	0.4pt
`\footrulewidth`	0 pt

在 `book.cls` 中，页眉的文字被转变成大写形式。

9 深入理解 LaTeX 的标记

通常，对于 `book` 或者 `report` 来说，你可能会把章节的信息反映在页眉上（对于单面打印可能只需要章次的信息），以及对于 `article` 文档类的节和子节（对于单面打印只需要节次的信息）。LaTeX 使用标记（mark）的机制来记录章节信息，在 LaTeX *Companion* 4.3.1节中有详细描述。

有两种方法可以改变高层的或低层[4] 的章节信息，下面两个命令 `\leftmark`（高层）和 `\rightmark`（低层）记录了 LaTeX 所要处理的信息。你可以如 8 节中描述的那样直接使用这两

[4]译者：原文为 `higher-level lower-level` 是相对于章节的层次结构而言的，这里翻译得不到位（有点像法轮功术语）

个命令。

命令 \leftmark 记录了页面上最近一次 \markboth 命令左边的参数，而 \rightmark 则记录了页面上第一次 \markboth 命令右边的参数，或者是页面上第一次 \markright命令的参数值。如果当前页面没有标记，则两者都维持前面的页面中的值保持不变。

你可以通过重新定义 \chaptermark, \sectionmark 和 \subsectionmark 命令[5] 来改变章和节或者节以及子节的信息。\pagestyle{fancy} 将会设置这些定义的默认值，因此你必须在这之后加入这些定义。

举个章次的例子，共有三个部分：

- 宏 \thechapter 所显示的章号（比如2）
- 宏 \chaptername 为章的名字（在英文中为 Chapter）
- 标题，包含在 \chaptermark 中.

图 3 显示了 “Chapter 2. Do it now” 标题的不同显示（最后一个例子比较适合非英语语言）。行末的 % 号用于去掉不需要的空格。通常你将会继续写下去而去掉这些 % 号[6]

对于低层的节次处理，也可以同样使用命令 \markright。

如果 “Section 2.2. 第一步” 为当前小节，则

```
\renewcommand{\sectionmark}[1]{\markright{\thesection.\ #1}}
```

将给出效果“2.2. 第一步”

重新定义 \chaptermark 和 \sectionmark 也未必可以取消所有的大写。比如参考文献的标题就为BIBLIOGRAPHY，因为在 \thebibliography 定义中早就调用了 \MakeUppercase。对于INDEX 也是同样。如果你不想重新定义这些命令，那么可以通过命令 \nouppercase 来使 fancyhdr 小写所有页眉和页脚的内容。注意，这种做法很有可能同其他东西发生冲突，比如页眉中大写的罗马数字，因此有必要谨慎使用。事实上，该命令通过将参数传递给一个 \MakeUppercase 和 \uppercase 被改变为空操作的环境来实现的。

```
\lhead{\nouppercase{\rightmark}}
\rhead{\nouppercase{\leftmark}}
```

LaTeX 的标记机制对于章（总是从新一页开始）和节（有相当长度），其工作良好。但对于短小的节次及子节就不那么理想了。问题出在 LaTeX 身上，而非fancyhdr。

举个例子来看，我们定义一种页面布局使得 \leftmark 由节生成而 \rightmark 由子节生成（作为article 文档类的默认设置）。如果页面由短小的节次组成，则

Section 1.
subsection 1.1
subsection 1.2
Section 2.

因为 leftmark 记录了最近一次页面上出现的标记，所以它的值现在为“Section 2.”，而 rightmark 则是“subsection 1.1”，因为它记录的是页面上第一次出现的标记。因此页眉上将同

[5]对于paragraph 以及 subparagraph 也有相同的命令，但很少有用。

[6]\MakeUppercase 命令在 LaTeX 2ε 中用于生成大写的文本，而在 LaTeX2.09版本中用的是 \uppercase。区别在于 \MakeUppercase 可以处理非 ASCII 字符。fancyhdr定义 \MakeUppercase 为 \uppercase 的别名如果事先没有定义。

代码:	打印出:

`\renewcommand{\chaptermark}[1]{%`
`\markboth{\chaptername`
`\ \thechapter.\ #1}{}}`
	Chapter 2. Do it now

`\renewcommand{\chaptermark}[1]{%`
`\markboth{\MakeUppercase{%`
`\chaptername}\ \thechapter.%`
`\ #1}{}}`
	CHAPTER 2. Do it now

`\renewcommand{\chaptermark}[1]{%`
`\markboth{\MakeUppercase{%`
`\chaptername\ \thechapter.%`
`\ #1}}{}}`
	CHAPTER 2. DO IT NOW

`\renewcommand{\chaptermark}[1]{%` `\markboth{#1}{}}`	Do it now
`\renewcommand{\chaptermark}[1]{%` `\markboth{\thechapter.\ #1}{}}`	2. Do it now

`\renewcommand{\chaptermark}[1]{%`
`\markboth{\thechapter.%`
`\ \chaptername.\ #1}{}}`
	2. Chapter. Do it now

图 3: 不同标记的变化

时出现 Section 2 和 Subsection 1.1，不是很美观。一个解决方法就是只用 `\rightmark` 而且重新定义 `\sectionmark` 。当然 LaTeX 命令 `\firstleftmark` 也是个不错的选择（参考 18 节中的`extramarks` 宏包）。

另外一个标准 LaTeX 文档类中标记的问题出在高层的章节命令如 `\chapter` 会调用 `\markboth` 而留空右边的参数。这意味着一章的第一页（或者是 article 文档类的第一页）中，`\rightmark` 将为空值。如果这种处理方法有违你的本意，那么你只好自己动手重新赋值给 `\rightmark`，或者重新写一遍 `\chaptermark`（或 `\sectionmark`）的定义，使得 `\markboth` 有两个合适的参数。

最后还要提醒你的是，`*` 形式的 `\chapter` 等命令不会调用标记相关的命令，因此如果你想要在前言中插入页眉或页脚，那么你就得自己加入 `\markboth` 命令，因为前言并不会计算入章节号，也不会被列入目录中。

```
\chapter*{Preface\markboth{Preface}{}}
```

把 `\markboth` 放在 `\chapter*` 中是因为这样可以确保不会因为分页的关系而丢失标记信息。当然，在 `\chapter*` 中不这样做并不成问题，因为 `\chapter*` 总是会另起一页，一般情况下也不会在标题之后就分页。但对于 `\section*` 就得注意一下了，千万不要这么写：

```
\section*{Preface}
\markboth{Preface}{}
```

因为很可能分页就发生在这两条命令中间。

10 字典样式的页眉

字典或类似的出版物通常会将第一个单词作为页眉，或者是第一个单词和最后一个单词。通过 fancyhdr 和 LaTeX 的 mark 机制我们可以很容易的做到这一点。当然，用于字典样式的标记机制并不适合章节信息，因此如果出现混排的情况，你需要重新定义 \chaptermark 以及 \sectionmark。

```
\renewcommand{\chaptermark}[1]{}
\renewcommand{\sectionmark}[1]{}
```

现在你可以用 \markboth{#1}{#1} 来记录字典中的每一项，然后通过 \rightmark 获得第一项而 \leftmark 获得最后一项。

如果你想使用类似 firstword–lastword 的页眉，那么最好在一页上只出现一个单词的情况下让它变成 firstworld 的形式。在这种情况下，必须要对两者作比较，但是 TeX 的标记比较奇怪，你不能简单地使用 \if 来作比较，所幸的是，ifthen 宏包可以。

```
\newcommand{\mymarks}{
  \ifthenelse{\equal{\leftmark}{\rightmark}}
    {\rightmark} % if equal
    {\rightmark--\leftmark}} % if not equal
\fancyhead[LE,RO]{\mymarks}
\fancyhead[LO,RE]{\thepage}
```

字典经常使用两栏的格式，不幸的是，LaTeX 的两栏选项有点小 bug，它会引起标记信息的丢失。如果你使用 fix2col[7]，就可以解决这个小问题[8]。或者你可以使用图 4 中的代码。

11 Fancy 布局

你可以通过 \\ 命令来生成一个多行的条目，也可以在其中使用命令 \vspace 来获得多一点的空隙。需要注意的是，这样做将会增加 页眉（\headheight）和页脚（\footskip）的高度，很有可能你就会得到一个 “Overfull \vbox ... has occurred while \output is active” 的错误[9]。想要获得更详细的资料，请参考 LaTeX *Companion* 第 4.1 节。

例如，下面的代码将一节的标题以及子节的标题分成两行显示在右上角：

```
\documentclass{article}
\usepackage{fancyhdr}
\pagestyle{fancy}
\addtolength{\headheight}{\baselineskip}
```

[7]你可以从 CTAN 站点上的 tex-archive/macros/latex/contrib/supported/carlisle/ 获取。
[8]multicol 宏包使用了类似的方法
[9]如果你使用 11pt 或者 12pt 的字体，你也许不得不这么做，因为 LaTeX 默认的字体太小了。

```
% fixmarks.sty:
% Patch LaTeX's output routine to handle marks correctly with two columns.
% Joe Pallas <pallas@edu.stanford.neon>
% Corrected by Piet van Oostrum <piet@cs.ruu.nl> on Feb 5, 1993, Oct 5, 1994

\def\@outputdblcol{\if@firstcolumn \global\@firstcolumnfalse
% Remember the marks from the first column
    \global\setbox\@leftcolumn\copy\@outputbox
    \splitmaxdepth=\maxdimen \cbaddness=10000
    \setbox\@outputbox\vsplit\@outputbox to\maxdimen
    \xdef\@firstcoltopmark{\topmark}%
    \xdef\@firstcolfirstmark{\splitfirstmark}%
    \ifx\@firstcolfirstmark\empty\global\let\@setmarks\relax\else
      \gdef\@setmarks{\let\firstmark\@firstcolfirstmark
        \let\topmark\@firstcoltopmark}%
    \fi
% End of change
  \else \global\@firstcolumntrue
    \setbox\@outputbox\vbox{\hbox to\textwidth{\hbox to\columnwidth
      {\box\@leftcolumn \hss}\hfil \vrule width\columnseprule\hfil
       \hbox to\columnwidth{\box\@outputbox \hss}}}\@combinedblfloats
% Override current first and top with those of first column if necessary
       \@setmarks
% End of change
       \@outputpage \begingroup \@dblfloatplacement \@startdblcolumn
       \@whilesw\if@fcolmade \fi{\@outputpage\@startdblcolumn}\endgroup
    \fi}
```

图 4: 修改 twocolumn 中的标记

```
\renewcommand{\sectionmark}[1]{\markboth{#1}{}}
\renewcommand{\subsectionmark}[1]{\markright{#1}}
\rhead{\leftmark\\\rightmark}
```

你也可以对修饰线进行调整，比如下面的代码可以将修饰线变的较粗：

```
\renewcommand{\headrulewidth}{0.6pt}
```

你也可以使页脚的修饰线消失：

```
\renewcommand{\footrulewidth}{0pt}
```

修饰线本身其实是 \headrule 和 \footrule 两个宏构成。例如，假设你看腻了实线而想要一条由点构成的虚线，你可以通过重新定义\headrule 来实现：

```
\renewcommand{\headrule}{\vbox to 0pt{\hbox
    to\headwidth{\dotfill}\vss}}
```

另外还有一个参数可能对你有用：\footruleskip，它定义了页脚修饰线和页脚文本最上方之间的距离。缺省值为30% 的正常行距。当然，你可能会将它定义为很宽或很窄，不过记住用 \renewcommand 来重新定义哦。

12 两个书本的例子

这里给出了一种接近于 L. Lamport 的LaTeX 书的样式，在 Lamport 的样式中，页眉超出了正常的边缘。页眉和页脚的长度由 \headwidth 定义，通常是正文的宽度。你可以使它宽一点或窄一点，只要用命令 \setlength 或 \addtolength 来重新定义 \headwidth 的长度就可以了。为了让页眉扩展到边注区，可以在 \headwidth 的基础上加上 \marginparsep 和 \marginparwidth，这两个参数分别是边注区同正文之间的空隙宽度以及边注区本身的宽度，如下所示：

```
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
```

你必须在第一次调用 \pagestyle{fancy} 命令之后发出这些命令，因为这样可以让调整后的 \headwidth 成为默认值[10]。

下面是 Lamport 书中布局样式的定义：

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
\addtolength{\headwidth}{\marginparsep}
\addtolength{\headwidth}{\marginparwidth}
\renewcommand{\chaptermark}[1]{\markboth{#1}{}}
\renewcommand{\sectionmark}[1]{\markright{\thesection\ #1}}
\fancyhf{}
\fancyhead[LE,RO]{\bfseries\thepage}
\fancyhead[LO]{\bfseries\rightmark}
\fancyhead[RE]{\bfseries\leftmark}
\fancypagestyle{plain}{%
   \fancyhead{} % get rid of headers
   \renewcommand{\headrulewidth}{0pt} % and the line
}
```

注意到 \chaptermark 和 \sectionmark 命令已经被重新定义，新的定义去掉了章次的序号并保持一章的标题为原来的大小写，而不是全部大写。

我们拿 $\mathcal{AMS}$-LaTeX book 作为第二个例子。

一章开始的页面没有页眉也没有页脚，因此对每一章开始的那一页，我们作如下声明：

```
\thispagestyle{empty}
```

这样一来省去了重新定义 plain 的麻烦。

章节的标题其形式为：2. DO IT NOW，因此我们对 \chaptermark 和 \sectionmark 进行重新定义（参考 9 节）：

```
\renewcommand{\chaptermark}[1]%
   {\markboth{\MakeUppercase{\thechapter.\ #1}}{}}
```

[10] 在 fancyhdr 版本 2 中，你可以在此之前设置，不过建议还是放在后面。

```
\renewcommand{\sectionmark}[1]%
   {\markright{\MakeUppercase{\thesection.\ #1}}}
```

在偶数页中，页码在页眉的左方而一章的标题在右；对应的，在奇数页中，页码在页眉的右方而小节的标题在左。页眉中间空着，也没有页脚。

在页眉下面有一条 0.5pt 宽的修饰线，因此我们这样定义：

```
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
```

页眉中使用了 9pt 大小的 Helvetica 粗体。在 Sebastian Rahtz 的 PSNFSS 系统中，Helvetica 遵从 Karl Berry 的字体短名称 phv，因此选择字体如下所示：

```
\fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont
```

（参考 LaTeX *Companion* 中第 7.6.1 和11.9.1 节）

定义简短命令：

```
\newcommand{\helv}{%
   \fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont}
```

现在可以进行页面布局了：

```
\documentclass{book}
\usepackage{fancyhdr}
\pagestyle{fancy}
\renewcommand{\chaptermark}[1]%
   {\markboth{\MakeUppercase{\thechapter.\ #1}}{}}
\renewcommand{\sectionmark}[1]%
   {\markright{\MakeUppercase{\thesection.\ #1}}}
\renewcommand{\headrulewidth}{0.5pt}
\renewcommand{\footrulewidth}{0pt}
\newcommand{\helv}{%
   \fontfamily{phv}\fontseries{b}\fontsize{9}{11}\selectfont}
\fancyhf{}
\fancyhead[LE,RO]{\helv \thepage}
\fancyhead[LO]{\helv \rightmark}
\fancyhead[RE]{\helv \leftmark}
```

13 浮动页面的特殊布局

人们希望一些浮动页面（就是只包含浮动对象的页面）可以有特殊的页面布局。因为这些浮动页面往往由 LaTeX 自动生成，因此用户无法对其进行控制，对浮动页面使用 `\thispagestyle` 不会奏效，因为这条命令在改变浮动页面样式的同时，至少也会改变之前一页的样式。而在 fancyhdr 中，你可以这样控制页眉和页脚：

`\iffloatpage`{value for float page}{value for other pages}

你甚至可以使用该判断去掉浮动页面上方的修饰线：

```
\renewcommand{\headrulewidth}{\iffloatpage{0pt}{0.4pt}}
```

有时你可能需要对那些浮动对象位于顶部或者底部的页面进行样式上的改动，fancyhdr 提供了 `\iftopfloat` 和 `\ifbotfloat` 命令，使用方法同 `\iffloatpage`。

注意：在浮动对象中定义的标记对于 LaTeX 的输出过程来说并不可见，因此在浮动对象中定义标记没有用处。目前还没有什么方法可以让浮动对象的某些属性（比如图形的标题）直接出现在页眉或者页脚中。

14 那些空白的页面

在 `book` 文档类中，`openany` 选项没有给出的情况下，或者是在 `report` 文档类给出 `openright` 选项的情况下，一章总是开始于奇数页，部分情况下会引起前面的一页为空白。有些人喜欢让这一页完全空白（也就是既没有页眉也没有页脚），`\thispagestyle` 对此无能为力，因为这条命令只可能在前面那一页中出现，这样做的后果是同时改变了前面一页的页面样式。但实际上，要想达到这样的效果并不难：

```
\clearpage{\pagestyle{empty}\cleardoublepage}
```

我们看到 `\pagestyle{empty}` 被放在了大括号之内，因此它影响且只影响到了由命令 `\cleardoublepage` 生成的页面样式。当然，你可以将上面这条命令放在自己定义的命令中，你也可以让每一章开始的时候自动生成，甚至在这些空白页面上放点什么文字，这时候，你必须重新定义 `\cleardoublepage` 命令：

```
\makeatletter
\def\cleardoublepage{\clearpage\if@twoside \ifodd\c@page\else
  \hbox{}
  \vspace*{\fill}
  \begin{center}
    This page intentionally contains only this sentence.
  \end{center}
  \vspace{\fill}
  \thispagestyle{empty}
  \newpage
  \if@twocolumn\hbox{}\newpage\fi\fi\fi}
\makeatother
```

15 N of M 样式的页码

一些文档的作者喜欢页码以 n of m 的样式出现，其中 m 为总页数而 n 为当前页数。有一个宏包 `nofm.sty` 可以做到这一点，但它的一些版本有点缺陷，大部分时候在 fancyhdr 中工作不正常，因为该宏包会修改整个页面的布局。在 LaTeX 2_ε 中有另外一个宏包 `lastpage` 可以在 fancyhdr 下使用：

```
\usepackage{lastpage}
...
```

```
\cfoot{\thepage\ of \pageref{LastPage}}
```

如果你还在使用 LaTeX2.09 而且不能升级到 LaTeX 2_ε，你可以使用同 LaTeX2.09 兼容的宏包 `lastpage209.sty`，它采用了如下定义：

```
\let\origenddocument=\enddocument
\def\enddocument{\clearpage\if@filesw
  {\addtocounter{page}{-1} \immediate\write\@mainaux
    {\string\newlabel{LastPage}{{}{\thepage}}}}\origenddocument}
```

`LastPage` 标签的值可以用来改变文档最后一页的页眉和页脚。比如除了最后一页，你想在所有奇数页的页脚中包含文字 "please turn over"。你可以这样做[11]：

```
\usepackage{lastpage}
\usepackage{ifthen}
...
\rfoot{\ifthenelse{\isodd{\value{page}} \and \not
    \value{page}=\pageref{LastPage}{please turn over}{}}
```

16 相对于章节的页码

在一些技术性文章中，页码通常为 2–10 的形式，第一个数字为章次，而第二个数字则是该页相对于本章开始的页数。当然，有些时候也可以相对于小节。chappg 宏包可以帮助你得到这种效果，如果你想要做点修改，比如中间用 "·" 而不是 "–"，那么你就得拷贝一份然后进行修改。

该宏包重新定义 `\thepage` 为 `\arabic{chapter}-\arabic{page}`。不幸的是，对于附录，这样的定义给出的是附录的序号而不是字母。更好的定义应该是 `\thechapter-\arabic{page}`，不过你可以在 `\usepackage{chappg}` 之后修改 `\thepage` 的定义。这个宏包所做的另外一件事情是在每一章开始之后将页数重置为 1。

对于前面所说的 "m of n" 的样式和上面描述的相对于章节的页码样式有着本质上的不同。m of n 的样式只能出现在页眉和页脚中，而不会出现在诸如目录、索引或者参照（如参照 xx 页）中。因此不需要改变 `\thepage` 的定义。而 "2–10" 的页码，必须在所有的引用中保持这种样式，因此它必须得重新定义 `\thepage`。

17 什么时候改变页眉和页脚的定义

有些时候你可能会在文档的中间想要改变页眉或者页脚的样式。有些修改可以通过标记的机制来完成，正如 9 和 18 节中所述。但一些时候，你需要做比较大的改动，比如改变页码样式，从罗马数字改成阿拉伯数字，或者是 改变 fancyhdr 中的一些变量，再有完全改变页面的样式等。这时候，你往往会发现这些改动比你预料的要早。通常，上面的改动将会马上奏效，也就是对当前正在处理的页面。如果你想要改变下一页的样式，你必须确保这一页已经完成。大部分时候，你可以在作出改动之前发出一条 `\clearpage` 命令，如果出于某种原因不能这样做，你可以使用宏包 `afterpage`：

[11]这需要一个版本较新的 `ifthen` 宏包。

`\afterpage{\lhead{new value}}` or `\afterpage{\pagenumbering{roman}`.

你不可以在 `\afterpage` 中使用 `\pagestyle` 命令，因为在 `\afterpage` 中所做的改变都是局部的。而 `\pagenumbering` 和 `fancyhdr` 命令所做的是全局的改动，因此可以正常工作，`\thispagestyle` 命令也可以。

需要注意的是，尽管 `fancyhdr` 的命令如 `\fancyhead` 可以立即起效，但这并不意味着这些命令中用到任何 "变量" 在此之后会立即被赋值。比如 `\fancyfoot[C]{\thepage}` 命令中给出的页码不是这条命令所在页的页码，而是当构建页脚时所在页的页码。当然，这只是对于页码来说，对于其他一些命令，可能这不是你想看到的效果。

因此，如果你要排版这样一本书，其中每个章节由不同的作者撰写，而且希望作者的名字位于左下脚，你可以给出下面的命令：

```
\newcommand{\TheAuthor}{}
\newcommand{\Author}[1]{\renewcommand{\TheAuthor}{#1}}
\lfoot{\TheAuthor}
```

然后在每一章的开始使用 `\Author{Real Name}`。但假设在页面还没有结束的时候就已经改变了作者的名字，那么该页左下脚就会出现错误的作者名。比方说在 `\chapter` 命令之前改变作者名字，而不是在之后。另外一种可能的原因是 TEX 的输出过程提前处理了命令，这就是说可能一些本该在后一页中输出某些文字的命令早就被执行了，详细例子参考下一节。

18 由文本引入的页眉和页脚

我们已经看到了如何利用 LATEX 的标记机制来使文章中的一些信息出现在页眉或者页脚上。标记机制也是得到这种功能的唯一可靠方法。这是因为 LATEX 可能会在分页之前预先处理一部分后面的内容[12]。

有些时候紧靠 LATEX 提供的两个标记还不够，请看下面的例子：

如果分页不得不在一道习题的解答中间进行，那么我情愿在前一页的底部标出 "(Continued on next page...)"，而在下一页的头上标出 "(Continued...)"。

你很难让 LATEX 的标记机制在记录章节信息的同时完成这样的工作。

图 5 中的代码所构成的宏包可以给你两个额外的标记来完成这样的功能[13]。

下面是该代码的使用范例：

```
\usepackage{extramarks}
...
\pagestyle{fancy}
\lhead{\firstxmark}
\rfoot{\lastxmark}
...
\extramarks{}{Continued on next page\ldots}
\ziti{C} 一些可能会越过页边界的段落文字\ldots
```

[12]译者：以决定最佳分页位置，参考 The TEXbook 第15章。

[13]在我写了这个宏包以后，我发现了一个 `secret.sty` 的文件也可以完成类似的事情，就是在重要的段落跨越页边界的时候进行标记。但它是通过修改输出过程的代码来完成此项工作的。

```
% extramarks.sty
\def\@leftmark#1#2#3#4{#1}
\def\@rightmark#1#2#3#4{#2}

\def\markboth#1#2{{\def\protect{\noexpand\protect\noexpand}
     \let\label\relax \let\index\relax \let\glossary\relax
     \expandafter\@markboth\@themark{#1}{#2}
     \mark{\@themark}}\if@nobreak\ifvmode\nobreak\fi\fi}
\def\markright#1{{\def\protect{\noexpand\protect\noexpand}
     \let\label\relax \let\index\relax \let\glossary\relax
     \expandafter\@markright\@themark
     {#1}\mark{\@themark}}\if@nobreak\ifvmode\nobreak\fi\fi}
\def\@markright#1#2#3#4#5{\gdef\@themark{{#1}{#5}{#3}{#4}}}
\def\@markboth#1#2#3#4#5#6{\gdef\@themark{{#5}{#6}{#3}{#4}}}
\def\leftmark{\expandafter\@leftmark\botmark{}{}{}{}}
\def\rightmark{\expandafter\@rightmark\firstmark{}{}{}{}}
\def\firstleftmark{\expandafter\@leftmark\firstmark{}{}{}{}}
\def\lastrightmark{\expandafter\@rightmark\botmark{}{}{}{}}

\def\@themark{{}{}{}{}}

\def\extramarks#1#2{{\def\protect{\noexpand\protect\noexpand}
     \let\label\relax \let\index\relax \let\glossary\relax
     \expandafter\@markextra\@themark{#1}{#2}
     \mark{\@themark}}\if@nobreak\ifvmode\nobreak\fi\fi}
\def\@markextra#1#2#3#4#5#6{\gdef\@themark{{#1}{#2}{#5}{#6}}}
\def\firstxmark{\expandafter\@firstxmark\firstmark{}{}{}{}}
\def\topxmark{\expandafter\@firstxmark\topmark{}{}{}{}}
\def\lastxmark{\expandafter\@lastxmark\botmark{}{}{}{}}
\def\@firstxmark#1#2#3#4{#3}
\def\@lastxmark#1#2#3#4{#4}
```

图 5: 在 LaTeX 中获取额外标记的代码

```
\extramarks{Continued\ldots}{}
```

注意 `\extramarks` 命令必须同与之相关联的文字紧紧相邻，也就是说不要留有空行（即段落的边缘）。否则分页可能在该边缘发生，而 extramarks 将出现在错误的页面上。

页面布局时你可以有两个额外的标记，如果给出命令 `\extramarks{`m_1`}{`m_2`}` 的话，`\firstxmark` 给你页面上出现的第一个m_1 标记值，`\lastxmark` 给你最后一个m_2 标记值。同时也提供了两个命令 `\firstleftmark` 和 `\lastrightmark` 作为标准 LaTeX 标记的补充。

为了强调一下标记机制是这类问题的正确解决方法，我们举一个不正确的 “解决” 方法[14]：

```
\lhead{Continued\ldots}
\rfoot{Continued on next page\ldots}
```

[14]实际上还有另外一种方法，但需要 LaTeX 编译两次，就是借助 `\label` ，在文本的前后插入标签，然后同 `\pageref` 做比较。

```
\ziti{C} 一些可能会越过页边界的段落文字\ldots
\lhead{}
\rfoot{}
```

你可能会不由自主地想到，如果 TEX 决定在文字中间分页，那么第一对 \lhead 和 \rfoot 有效，而在分页之后则第二对起作用。然而这种想法却是错误的，因为 TEX 在考虑分页 时已经处理了整个段落（包括最后一对定义），因此在分页时，不管是在段落中间还是在该段落之后，最后一对定义将起决定性作用。你可能会想到在最后一对定义和上面的段落之间增加一个段落分隔。但这样还是没用，因为当 TEX 决定在这个分隔处分页时，你显然不想让后面一页按照第一对定义来显示页眉和页脚。事实上，标记机制的发明就是为了解决这些问题。

在上面的例子中，文字 “Continued” 出现在页眉上，不过最好是放在边上，只要将它放在相对于页眉的一个固定位置上。在 plain TEX 中，你可以使用如 \hbox to 0pt、\vbox to 0pt、\hskip、\vskip、\hss 和 \vss 的组合来达到这种效果。所幸的是，LATEX 的 picture 环境让这件工作变得更加方便。在不影响正常页眉的情况下，我们把文本放到一个零大小 picture 中。这实际上是比较好的一种通用方法，可以把东西放在页面的固定位置上。你也可以同时使用正常的页眉。参考 20 节，那里有使用该技术的另一个例子。

```
\lhead{\setlength{\unitlength}{\baselineskip}%
\begin{picture}(0,0)
  \put(-2,-3){\makebox(0,0)[r]{\firstxmark}}
\end{picture}\leftmark}
```

这个方法当然也适用于页脚。不过确保将 picture 放在左侧条目的第一项，以及右侧条目的最后一项[15]。

最后，你想把 “(Continued on next page...)” 放在文本中而不是页眉或者页边上，这样你恐怕就需要 afterpage.sty 宏包了，我们也为之设计了单独的环境：

```
\newenvironment{continued}{\par
  \extramarks{}{Continued on next page\ldots}
  \afterpage{\noindent\firstxmark\vspace{1ex}}
  }{\extramarks{(Continued\ldots)}{}\par}
```

尽管在页面布局过程外面使用 \firstxmark 有点危险，不过在 \afterpage 下似乎还没有问题。如果你还需要记录页面上的一些信息，你必须将标记的信息记录在你自己的变量中，并且放入 fancyhdr 的一些定义中。比方说，你想在某段文字之后加入点什么东西，你可以使用下面的代码：

```
\newcommand{\mysaved}{}

\newenvironment{continued}{\par
  \extramarks{}{Continued on next page\ldots}
  }{\extramarks{(Continued\ldots)}{}\par\vspace{1ex}\mysaved}
\lhead{\leftmark}
\chead{\ifthenelse{\equal{\lastxmark}{}}
```

[15]译者：作者的本意可能是说不要放在左页面底部和右页面顶部的地方，因为那样不用 “未完待续” 也一目了然，条目可能是针对字典样式二来。

```
{\gdef\mysaved{}}
{\gdef\mysaved{\noindent[Continued from previous page]}}}
```

如果你想要获取已保存文本中的某个标记或者其他变动的信息，你应该使用 `\xdef` 而不是 `\gdef`。

19 小电影

如果你在每一页的同一个地方放置一些彼此之间只有微小差别的图片，那么当你快速翻页的时候就会得到一种小电影的效果。使用 fancyhdr，你可以轻松达到这种效果。为简单起见，我们假设每张图片都是 postscript(EPS) 文件，文件名为 `pic`⟨n⟩`.ps` 的格式，⟨n⟩ 为页码，我们使用 `graphics` 或者 `graphicx` 宏包[16]。

下面的代码把小电影放在了右侧页面的底部：

```
\rfoot{\setlength{\unitlength}{1mm}
  \begin{picture}(0,0)
    \put(5,0){\includegraphics{pic\thepage.ps}}
  \end{picture}}
```

注意，`\unitlength` 必须设置为局部有效，否则会影响正文中的其他设置。

20 书边索引

一些铁路指南或者昂贵的圣经中的有些具有所谓的*书边索引* [17]，也就是在页面边缘上标识章节所在位置的记号。你可以通过在页边缘打印上一黑块达到类似的效果。该黑块的垂直位置由章节号或其他计数器决定。因为这些位置和页面其他内容无关，因此我们在页眉中把它作为零大小的图片来处理，正像前面一节中所描述的那样。

当然我们必须考虑到双面打印的可能，因此我们要把这些黑块放在对应的地方。有时候，不得不手动调整这些黑块使之在垂直方向上正确分隔。比如有一份 12 节的文档，我便设置块与块之间相隔 18 mm，也就是 9 mm 的块和 9 mm 的空白。为了避免繁琐的计算，在 `picture` 环境中，`\unitlength` 被设置为 18 mm。页码被打印在边上，而这些黑块则附在其上。在本例中，节次的序号用于决定这些块的位置，但你可以用其他数值来代替它。参考图 6 的大致效果，图7 包含了它的代码。

21 浮动对象的放置

浮动对象是那些所谓浮动于文档其他部分之上的元素，标准的浮动对象包括表格和图片。但通过 float 宏包，你可以方便地创建你自己地浮动对象，比如算法啊什么的。大部分时候，float 宏包工作良好，令人满意，但有些时候 LaTeX 对于你给的指令太过固执，可能会产生不好的效果。在这一节中，我们将描述如何让 LaTeX 做你想要做的事情。当然也有可能在一些比较变态的例子中，我们无法使得 LaTeX 按照我们的方式工作。在下面的例子中，我们以图片为例，但

[16] 如果你正在使用较老版本的 LaTeX，你可以使用 `epsf` 或者 `epsfig` 宏包。
[17] 译者：原文 thumb index

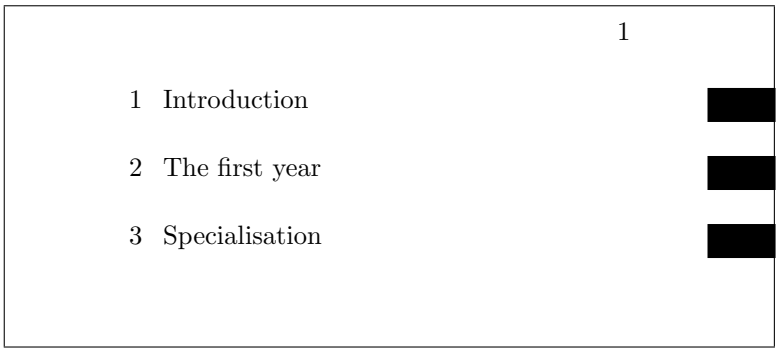


图 6: Thumb-index overview page

也适用于其他浮动对象。

通常遇到的有关浮动对象的问题不外乎：

1. 你想让某浮动对象位于页面的某个位置，但 LaTeX 将之挪了位置，通常是挪到下一页上。
2. 从某一点开始，LaTeX 将你所有的浮动对象挪到一章的尾部。
3. LaTeX 抱怨说 “Too many floats”。

在头两种情况下，你必须检查有没有给对 “placement” 参数，比如，`\begin{figure}[htp]` 指定了你的图片可以放在：要么原地（就是文本中命令给出的地方），或者是页面的顶部（也可能是命令给出的那一页），要么就是放在只含有浮动对象的单独一页上。你也可以指定 “b” 表明允许放在一页的底部。参数的顺序没有讲究，你也不能通过参数 `[bt]` 让 LaTeX 先尝试底部再试试顶部这样子来放置浮动对象。

如果 LaTeX 并没有将浮动对象放在你想要的地方，可能的原因有：

1. 浮动对象无法安排在当前页上，这种情况下，它可能会被挪到后一页甚至更后面的页面上。如果你即没有指定 `[t]` 也没有指定 `[b]`，LaTeX 会将之保存起来，直到有一页浮动页面具有足够的空间。因此不要只指定 `[h]`。如果你想有机会让 LaTeX 把它放到浮动页面上去，你也必须指定 `[p]`。
2. 放置的位置和 LaTeX 的浮动对象放置参数发生了冲突，这是很长剑的错误，也可以通过改变这些参数方便地纠正错误。这里是这些参数的列表：

Counters – 用 `\setcounter` 来更改		
`topnumber`	页面顶部最多可以放置的浮动对象数	2
`bottomnumber`	页面底部最多可以放置的浮动对象数	1
`totalnumber`	页面上最多可以放置的浮动对象数	3
另外一些 – 用 `\renewcommand` 来更改		
`\topfraction`	顶部浮动对象占整个页面的比例	0.7
`\bottomfraction`	底部浮动对象占整个页面的比例	0.3
`\textfraction`	页面上正文空间的最小比例	0.2
`\floatpagefraction`	浮动页面上浮动对象占用的最小比例	0.5

在双栏的文档中还有一些其他参数。

在最右边的一栏中是标准 LaTeX 类的默认值。其他文档类可能使用其他数值。你可以看到，默认情况下，如果浮动对象的高度超过页面高度的 30% 时，将不会被放置在页面底部。因此假设你指定了 `[hb]`，而且浮动对象比较高，那么它将会被挪到浮动页面上，但如果它的

```
\setlength{\unitlength}{18mm}
\newcommand{\blob}{\rule[-.2\unitlength]{2\unitlength}{.5\unitlength}}

\newcommand\rblob{\thepage
  \begin{picture}(0,0)
    \put(1,-\value{section}){\blob}
  \end{picture}}

\newcommand\lblob{%
  \begin{picture}(0,0)
    \put(-3,-\value{section}){\blob}
  \end{picture}%
  \thepage}

\pagestyle{fancy}
\cfoot{}

\newcounter{line}
\newcommand{\secname}[1]{\addtocounter{line}{1}%
  \put(1,-\value{line}){\blob}
  \put(-7.5,-\value{line}){\Large \arabic{line}}
  \put(-7,-\value{line}){\Large #1}}

\newcommand{\overview}{\thepage
  \begin{picture}(0,0)
    \secname{Introduction}
    \secname{The first year}
    \secname{Specialisation}
...etc...
  \end{picture}}

\begin{document}
\fancyhead[R]{\overview}\mbox{}\newpage % This produces the overview page
\fancyhead[R]{} % Front matter may follow here
\clearpage
\fancyhead[RE]{\rightmark}
\fancyhead[RO]{\rblob}
\fancyhead[LE]{\lblob}
\fancyhead[LO]{{\leftmark}
...
```

图 7: Thumb-index code

高度小于页面高度的 50%，那么它将等待有足够的浮动对象可以放在浮动页面上并且满足 \floatpagefraction 参数。如果你有这种类型的情况，你可以通过改变参数来适应，比如：

```
\renewcommand{\textfraction}{0.05}
\renewcommand{\topfraction}{0.95}
\renewcommand{\bottomfraction}{0.95}
\renewcommand{\floatpagefraction}{0.35}
\setcounter{totalnumber}{5}
```

你必须注意不要把 \floatpagefraction 改得太小，否则你会得到很多浮动页面。

对于某个浮动对象，你可以让 LaTeX 忽略大部分这些参数，只要在位置参数中加入一个惊叹号（!），例如：

```
\begin{figure}[!htb]
```

如果位置参数中含有 “t”，那么浮动对象可能在之前被输出（当然，至少是在同一页面上），这是 LaTeX 正常处理方式，不过有些人就是不喜欢。有很多方法可以避免 LaTeX 的这种正常方式：

1. 从位置参数中去掉 “t” 当然可以，不过不是好方法，因为你可能想让改浮动对象出现在下一页的顶部。
2. 使用 flafter 宏包，它禁止浮动对象出现在引用之前。
3. 使用命令 \suppressfloats[t] [18]，该命令将出现在当前页顶部的浮动对象挪到下一页。这也可以通过 [b] 或者对该页上所有浮动对象不指定参数来完成。

如果在你做了所有尝试之后，LaTeX 还是把你的浮动对象挪到文档最后或者章节的最后，你可以试着插入一些 \clearpage，这样没被处理的浮动对象就可以被放置在新的一页上。如果不想插入不恰当的分页位置，你可以使用 afterpage 宏包，正如下面的代码，

```
\afterpage{clearpage}
```

这样一来，在该页结束之后将新开一页来安排没有被处理的浮动对象。在一些比较变态的情况下，afterpage 也许会产生一些比较奇怪的结果。

最后，如果你想让浮动对象呆在原地而不要让 LaTeX 挪来挪去，你可以使用 float 宏包，只要在导言区中敲入命令：\restylefloat{figure}。现在你就可以在位置参数的地方指定 [H]，表示 “只准在这里”。不过，这样可能会因起非预期的分页位置[19]。如果你想避免非预期的分页位置，也就是说让 LaTeX 只在浮动对象不能安排在当前位置的情况下对其进行挪动，那么可以这样用

```
\afterpage{\clearpage \begin{figure}[H] ... \end{figure}}
```

。

有时候LaTeX 会抱怨 “Too many floats”，这通常是由以上的原因之一造成，也就是 LaTeX 收集了太多不能马上被放置的浮动对象的时候。上面的一些解决方法，尤其是 \clearpage 通常比较有用。在另一些情况下，浮动对象实在太多而超过了 LaTeX 用于保存它们的 “boxes” 的数量上限。这种时候，morefloats 可以增加其上限，如果这个上限还不够的话，你最好保存该文件的一份拷贝，然后再对其进行编辑，不过大多数情况下，你最终不得不重新修改你的文档。

[18] 该命令以及 “!” 位置参数在 LaTeX2.09 中并没有定义。

[19] 过去曾有过 here.sty 可以得到同样的效果，不过它和 LaTeX 2ε 不兼容

22 多页的浮动对象

LaTeX 的浮动对象不能越过页边界。有些时候，你可能有某个表或某张图片甚至一整页都放不下。最简单的方法是将之分成几张表或几张图使之能够在一页上放下，但这样做会有一些非预期的效果：

- 你在那里将之分离？通常回答这个问题对于表格比对于图片来说更困难。
- 你如何保证他们将被连续安放？
- 你不想在图片或者表格列表中有多余的表项。

尽管这些问题还没有在所有的情况下被完全解决，这里有一些建议：

§22.1 表格

对于那些长度大于页面长度的表格，你可以使用 `longtable` 宏包。这个宏包定义了一个 `longtable` 的环境，基本上是 `table` 和 `tabular` 环境的融合。它的语法同 `tabular` 环境很相像，但也加入了一些 `table` 的特性，比如标题。`longtables` 将在一页中放不下的时候自动分裂，而且在给出标题后，表格列表中也只保留一条表目。但它并不浮动，也不能在浮动环境中使用，这也意味着在 `longtable` 之前可能定义的 `table` 浮动对象有可能越过它，序号也可能会被搞乱。另一个问题是，`longtable` 可能出现在页面的底部，这种效果并不舒服。如果你想让它出现在页面的顶部，你可以将之放在 `\afterpage` 命令（使用 `afterpage` 宏包）中。因为 `longtable` 很大，你最好将之放在一个单独文件中，然后在 `\afterpage` 命令中用 `\input` 包含它。

```
\afterpage{\input{mytable}}
```

```
\afterpage{\clearpage\input{mytable}}
```

后一种方式有个好处就是大部分待处理的浮动对象会先被输出。

§22.2 图形

对于图形，没有对应的 `longfigure` 的解决方法，通常你只好亲自将它们分开。大部分时候，这还不是什么大问题，关键在于如何让它们被连续安放，以及如何在图形列表中保持单条记录。

在将图形分割成几块并且每块都放在 `figure` 环境中后，为了让它们保持连续，最好使用 `[p]` 位置参数，这样它们将会被放置在浮动页面上。因为图片比页面大，所以通常这是可行的方法。第一幅图片的标题 `\caption` 将被设置，而接下去的图片则不要设置。如果你想得到类似于图片标题的效果，用普通的文本而非 `\caption`，这样在图形列表中只保留了一条记录。也可以首先使用一条 `\clearpage` 命令，就象在 `longtable` 中一样，然后将之加入 `\afterpage` 命令中，如下：

```
\afterpage{\clearpage\input{myfigure}}
```

where `myfigure.tex` contains:

```
\begin{figure}[p]
\includegraphics{myfig1.eps}
\caption{This is a multipage figure}
\label{fig:xxx}
```

```
\end{figure}
\begin{figure}[p]
\includegraphics{myfig2.eps}
\begin{center}
Figure~\ref{fig:xxx} (continued)
\end{center}
\end{figure}
```

必须确保图形的最后一部分足够大，否则 LaTeX 可能会先将之收集起来，在达到足够的浮动对象之后再行输出。一方面，可以将图形分割地足够大（比方加入一些 `\vspace`），另一方面也可以调整 `\floatpagefraction` 参数来完成。

如果希望这些跨越多页的图形从左页（即偶数页）开始，你可以在 `\afterpage` 命令中加入一些测试（可以使用 `ifthen` 宏包）：

```
\afterpage{\clearpage
\ifthenelse{\isodd{\value{page}}{\afterpage{\input{myfigure}}} % odd page
    {\input{myfigure}}}} % even page
```

当然，如果被跳过的页面上依然有过多的浮动对象，这个方法还是会失效。

23 联系信息

Piet van Oostrum
Dept. of Computer Science
Utrecht University
P.O. Box 80.089
3508 TB Utrecht, The Netherlands
Telephone: +31 30 2532180 Telefax: +31 30 2513791
E-mail: piet@cs.uu.nl
WWW: http://www.cs.uu.nl/people/piet